Using MaBoSS with pyMaBoSS via CoLoMoTo jupyter notebook

Vincent Noël[1]

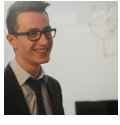
[1]Computational Systems Biology of Cancer,
Institut Curie, INSERM U900, Mines ParisTech, PSL Research University, Paris

Computational Systems Biology for Complex Human Disease
from static to dynamic representations of disease mechanisms

December 7th, 2022

Using MaBoSS with pyMaBoSS

- Initially developped by Nicolas Levy
- Maintained by Aurelien Naldi, Loic Pauleve, me
  https://github.com/colomoto/pyMaBoSS
- Available on Pypi:
  $ pip install maboss
- Available on Conda:
  $ conda install -c colomoto pymaboss


Nicolas Levy


Aurelien Naldi


Loïc Paulevé


Vincent Noël

Using MaBoSS with pyMaBoSS

› Loading a model

```
In [1]: import maboss
        model = maboss.load("metastasis.bnd", "metastasis.cfg")
```

```
In [ ]: sbml_model = maboss.loadSBML("Cohen.sbml", "metastasis.cfg")
```

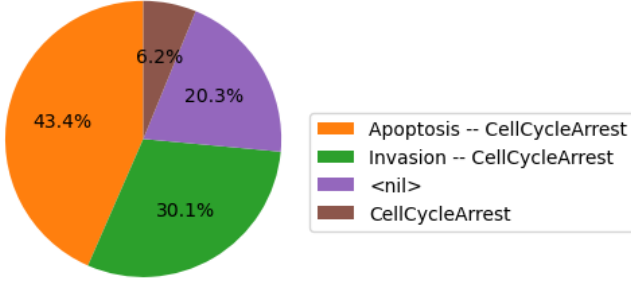
```
In [ ]: bnet_model = maboss.loadBNet("Cohen.bnet", "metastasis.cfg")
```

› MaBoSS is directly compatible with SBML, BNet, and MaBoSS proprietary format for network representation

Using MaBoSS with pyMaBoSS

› Running a simulation

```
import maboss
model = maboss.load("metastasis.bnd", "metastasis.cfg")
res = model.run()
res.plot_piechart()
```

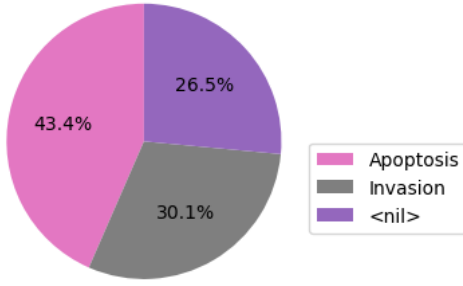


› A few lines of code to simulate the model and plot the final states distribution

Using MaBoSS with pyMaBoSS

› Changing output nodes

```
model.network.set_output(["Apoptosis", "Invasion"])
res = model.run()
res.plot_piechart()
```

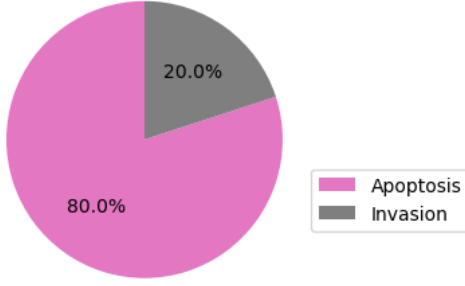


› Controlling which node is included in the results (Ex: remove CellCycleArrest)

Using MaBoSS with pyMaBoSS

- Changing initial states

```
model.network.set_istate("ECMicroenv", [0, 1])
model.network.set_istate("DNAdamage", [0, 1])
res = model.run()
res.plot_piechart()
```

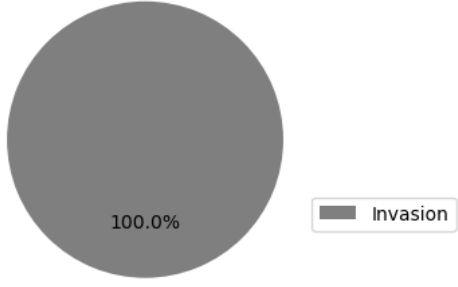


- Simulating with specific initial values

Using MaBoSS with pyMaBoSS

› Simulating mutations

```
model_mutant = model.copy()
model_mutant.mutate('NICD','ON')
model_mutant.mutate('p53','OFF')
res_mutant = model_mutant.run()
res_mutant.plot_piechart()
```

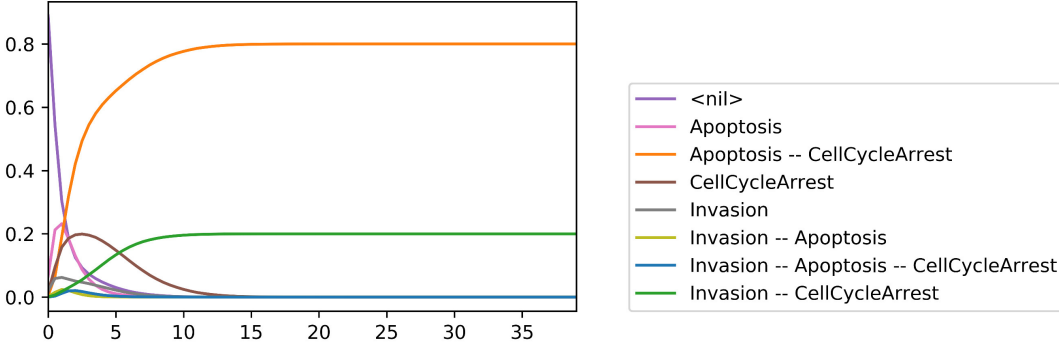


› Performing mutant simulations

Using MaBoSS with pyMaBoSS

› Results : State probability distribution trajectories

```
res.plot_trajectory()
```

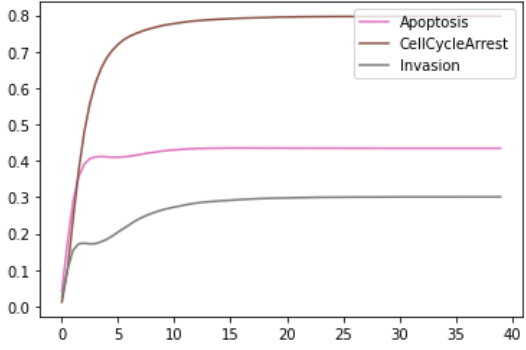


› Visualizing state probability trajectories

Using MaBoSS with pyMaBoSS

› Results : Nodes probability trajectories

```
res.plot_node_trajectory()
```



› Visualizing state probability trajectories

Using MaBoSS with pyMaBoSS

› Results : State probability distribution trajectories values

```
res.get_states_probtraj()
```

	<nil>	Apoptosis	Apoptosis -- CellCycleArrest	CellCycleArrest	Invasion	Invasion -- Apoptosis	Invasion -- Apoptosis -- CellCycleArrest	C
0.0	0.927211	0.036734	0.002495	0.008816	0.023224	0.001194	0.000086	
0.5	0.682001	0.124939	0.035559	0.056035	0.081564	0.012353	0.002428	
1.0	0.485872	0.146506	0.109373	0.103834	0.104372	0.022607	0.010071	
...	...	...	...	...	...	...	...	
38.0	0.202716	0.000000	0.434370	0.061990	0.000004	0.000000	0.000000	
38.5	0.202714	0.000000	0.434370	0.061996	0.000000	0.000000	0.000000	
39.0	0.202719	0.000000	0.434370	0.061991	0.000000	0.000000	0.000000	

› Getting the states probability trajectories as Panda dataframes

Using MaBoSS with pyMaBoSS

› Results : Nodes probability trajectories values

```
res.get_nodes_probtraj()
```

	Apoptosis	CellCycleArrest	Invasion
0.0	0.040509	0.011637	0.024744
0.5	0.175279	0.099143	0.101466
1.0	0.288557	0.240643	0.154415
...	...	...	...
38.0	0.434370	0.797280	0.300924
38.5	0.434370	0.797286	0.300920
39.0	0.434370	0.797281	0.300920

› Getting the node probability trajectories as Panda dataframes

Using MaBoSS with pyMaBoSS

› Results : Last state probability distributions

```
res.get_last_states_probtraj()
```

	<nil>	Apoptosis -- CellCycleArrest	CellCycleArrest	Invasion -- CellCycleArrest
39.0000	0.202719	0.43437	0.061991	0.30092

```
res.get_last_nodes_probtraj()
```

	Apoptosis	CellCycleArrest	Invasion
39.0000	0.43437	0.797281	0.30092

› Getting the last state probability distribution as Panda dataframes

Using MaBoSS with pyMaBoSS

› Sensitivity analysis : Double mutants

```
from maboss.pipelines import simulate_double_mutants
```

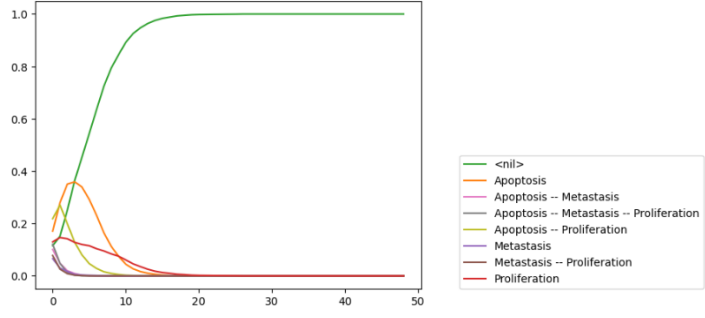
```
candidates_nodes = ["AKT", "ERK", "ROS"]
```

```
simulations = simulate_double_mutants(maboss_model, candidates_nodes , "OFF")
```

```
print(simulations)
```

```
{(('AKT', 'OFF'), ('ERK', 'OFF')): <maboss.result.Result object at 0x7f704b0fbd90>, (('AKT', 'OFF'), ('ROS', 'OFF')): <maboss.result.Result object at 0x7f7049587dc0>, (('ERK', 'OFF'), ('ROS', 'OFF')): <maboss.result.Result object at 0x7f70497748b0>}
```

```
simulations[(('AKT', 'OFF'), ('ERK', 'OFF'))].plot_trajectory()
```



Using MaBoSS with pyMaBoSS

› Sensitivity analysis : Filtering results

```
from maboss.pipelines import filter_sensitivity
```

```
res_filtered = filter_sensitivity(simulations, state='Proliferation', maximum=0)
```

```
res_filtered
```

```
{(('AKT', 'OFF'), ('ERK', 'OFF')): <maboss.result.Result at 0x7f704b0fbd90>,
 (('AKT', 'OFF'), ('ROS', 'OFF')): <maboss.result.Result at 0x7f7049587dc0>}
```

› Filtering sensitivity analysis results by phenotype

Using MaBoSS with pyMaBoSS

- Hands on
  - Load Montagud's model
  - Simulate and plot default model
  - Simulate and plot model with all initial values at zero
  - Simulate proliferative conditions
  - Simulate proliferative conditions with MYC_MAX inhibition

Using MaBoSS with pyMaBoSS

- Hands on
  - Simulate a batch in inhibitions
  - Filter inhibition with zero proliferation and report which ones
  - Simulate default conditions for all personalized models, and build a dataframe which all the final states. Report min/max apoptosis
  - Simulate proliferative conditions with MYC_MAX inhibition for all personalized models, build dataframe with all the final states
  - Filter patients with less than 10% of Proliferation and report them